

**PATENT APPLICATION
ATTORNEY DOCKET NO. SUN04-0550-EKL**

5

10

**METHOD AND APPARATUS FOR
DISPLAYING RELATED TWO-
DIMENSIONAL WINDOWS IN A THREE-
DIMENSIONAL DISPLAY MODEL**

15

**Inventors: Hideya Kawahara, Daniel J. Baigent, Brian D. Nitz,
Curtis J. Sasaki, Jonathan I. Schwartz, Hans E. Muller and Carl I. Gadener**

Related Application

20

[0001] This application is a continuation-in-part of, and hereby claims priority under 35 U.S.C. § 120 to, U.S. Patent Application Serial No. 10/663,609, entitled, "Method and Apparatus for Manipulating Two-Dimensional Windows Within a Three-Dimensional Display Model," by inventor Hideya Kawahara, filed 15 September 2003 (Attorney Docket No. SUN04-0195-EKL).

25

BACKGROUND

Field of the Invention

30

[0002] The present invention relates to user interfaces for computer systems. More specifically, the present invention relates to a method and an

apparatus for displaying related two-dimensional windows within a three-dimensional display model.

Related Art

5 **[0003]** Today, most personal computers and other high-end devices support window-based graphical user interfaces (GUIs), which were originally developed back in the 1980's. These window-based interfaces allow a user to manipulate windows through a pointing device (such as a mouse), in much the same way that pages can be manipulated on a desktop. However, because of
10 limitations on graphical processing power at the time windows were being developed, many of the design decisions for windows were made with computational efficiency in mind. In particular, window-based systems provide a very flat (two-dimensional) 2D user experience, and windows are typically manipulated using operations that keep modifications of display pixels to a
15 minimum. Even today's desktop environments like Microsoft Windows (distributed by the Microsoft Corporation of Redmond, Washington) include vestiges of design decisions made back then.

[0004] In recent years, because of increasing computational requirements of 3D applications, especially 3D games, the graphical processing power of
20 personal computers and other high-end devices has increased dramatically. For example, a middle range PC graphics card, the "GeForce2 GTS" distributed by the NVIDIA Corporation of Sunnyvale, California, provides a 3D rendering speed of 25 million polygon-per-second, and Microsoft's "Xbox" game console provides 125 million polygon-per-second. These numbers are significantly better than
25 those of high-end graphics workstation in the early 1990's, which cost tens of thousands (and even hundreds of thousands) of dollars.

[0005] As graphical processing power has increased in recent years, a number of 3D user interfaces have been developed. These 3D interfaces typically allow a user to navigate through and manipulate 3D objects. However, these 3D interfaces are mainly focused on exploiting 3D capabilities, while little attention
5 has been given to supporting existing, legacy window-based 2D applications within these 3D user interfaces.

[0006] If a 3D interface is to be commercially viable, it is crucial to be able to support the large existing base of legacy 2D applications. One of the problems that arises in trying to use 2D applications within a 3D interface is how
10 to arrange related 2D windows in an intuitive and convenient way within the 3D interface. Note that within a 3D interface, it is possible to indicate relationships between 2D windows through a large number of possible spatial relationships.

[0007] Hence, what needed is a method and an apparatus for displaying related 2D window-based applications within a 3D user interface.
15

SUMMARY

[0008] One embodiment of the present invention provides a system that facilitates displaying multiple two-dimensional (2D) windows with related content within a three-dimensional (3D) display model. The system starts by receiving a
20 command to display a first window within the 3D display model. In response to the command, the system displays the content of the first window on a first surface of a 3D object. Next, the system receives a command to display a second window within the 3D display model, wherein content of the second window is related to content of the first window. The system then displays content of the
25 second window on a second surface of the 3D object.

[0009] In a variation on this embodiment, the second surface of the 3D object is located on the opposite side of the 3D object from the first surface,

wherein only one of the first surface of the 3D object and the second surface of the 3D object is visible at any given time.

[0010] In a further variation, the system rotates the 3D object so that the second surface is visible.

5 [0011] In a variation on this embodiment, the system receives a command to display a third window within the 3D display model. In response to this command, the system displays content of the third window on a surface of a second 3D object, wherein the second 3D object is located in close proximity to the 3D object in the 3D display model.

10 [0012] In a further variation, the system receives a modal dialog related to the content of the first window, wherein the modal dialog must be responded to before any other action may be taken on an application. In order to display the modal dialog, the system rotates the 3D object so that the second surface is visible and the first surface is hidden, and displays the modal dialog on the second
15 surface.

[0013] In a further variation, when the modal dialog is displayed, the system rotates any related 3D objects so that related content on the surface of the related 3D objects is not visible until the modal dialog is acknowledged.

[0014] In a variation on this embodiment, the first window and the second
20 window are associated with different applications.

[0015] In a variation on this embodiment, upon receiving the command to display the second window, the system looks up an identifier for the second window in a lookup table that contains entries specifying relationships between windows. The system then determines if the second window is related to the first
25 window, and if so, displays content of the second window on the second surface of the 3D object. If the first and second windows are unrelated, the system

displays content of the second window on a surface of a distant 3D object, which is not located in close proximity to the 3D object in the 3D display model.

[0016] In a variation on this embodiment, the system receives a notification that the first window and the second window contain related content.

5 In response to this notification, the system creates an association between the first window and the second window in a lookup table.

[0017] In a variation on this embodiment, the 3D object is stacked on top of the second 3D object so that the second 3D object is obscured by the 3D object from the viewpoint of a user.

10 [0018] In a variation on this embodiment, the 3D object is translucent so that the second 3D object is visible through the 3D object.

BRIEF DESCRIPTION OF THE FIGURES

[0019] FIG. 1 illustrates a 3D display model with supporting components
15 in accordance with an embodiment of the present invention.

[0020] FIG. 2 presents a flow chart illustrating how input from a pointing device is communicated to an application associated with a window in a 3D display model in accordance with an embodiment of the present invention.

[0021] FIG. 3 presents a flow chart illustrating how input from a pointing
20 device causes objects to rotate around a viewpoint in the 3D display model in accordance with an embodiment of the present invention.

[0022] FIG. 4A illustrates an exemplary window in the 3D display model in accordance with an embodiment of the present invention.

[0023] FIG. 4B illustrates how the exemplary window is rotated to display
25 application information on the backside of the window in accordance with an embodiment of the present invention.

[0024] FIG. 4C presents a flow chart of the process of rotating a window in accordance with an embodiment of the present invention.

[0025] FIG. 5 illustrates a 3D object with multiple viewing surfaces in accordance with an embodiment of the present invention.

5 [0026] FIG. 6 presents a flow chart of the process of displaying a window in accordance with an embodiment of the present invention.

[0027] FIG. 7 presents a flow chart of the process of displaying a modal dialog in accordance with an embodiment of the present invention.

[0028] FIG. 8 illustrates object translucency in accordance with an
10 embodiment of the present invention.

[0029] Table 1 illustrates an exemplary lookup table in accordance with an embodiment of the present invention.

DETAILED DESCRIPTION

15 [0030] The following description is presented to enable any person skilled in the art to make and use the invention, and is provided in the context of a particular application and its requirements. Various modifications to the disclosed embodiments will be readily apparent to those skilled in the art, and the general principles defined herein may be applied to other embodiments and applications
20 without departing from the spirit and scope of the present invention. Thus, the present invention is not intended to be limited to the embodiments shown, but is to be accorded the widest scope consistent with the principles and features disclosed herein.

[0031] The data structures and code described in this detailed description
25 are typically stored on a computer readable storage medium, which may be any device or medium that can store code and/or data for use by a computer system. This includes, but is not limited to, magnetic and optical storage devices such as

disk drives, magnetic tape, CDs (compact discs) and DVDs (digital versatile discs or digital video discs), and computer instruction signals embodied in a transmission medium (with or without a carrier wave upon which the signals are modulated). For example, the transmission medium may include a
5 communications network, such as the Internet.

3D Display Model

[0032] FIG. 1 illustrates 3D display model 102 with supporting components in accordance with an embodiment of the present invention. More specifically, the top portion of FIG. 1 illustrates 3D display model 102, which
10 includes a number of 3D objects including window 108, window 110, and 3D object 111. Note that windows 108 and 110 are actually 3D objects which represent 2D windows. Hence, windows 108 and 110 can be moved and rotated within 3D display model 102, while they provide a 2D output and receive input
15 for associated 2D applications. 3D display model 102 can additionally include a background (which is not shown).

[0033] Windows 108 and 110 can be associated with a number of window attributes. For example, window 110 can include *x*, *y*, and *z* position attributes that specify the 3D position of the center of window 110 within 3D display model
20 102, as well as a rotation attributes that specify rotations of window 110 around horizontal and vertical axes. Window 110 can also be associated with scaling factor, translucency and shape attributes.

[0034] 3D objects within 3D display model 102 are viewed from a viewpoint 106 through a 2D display 104, which is represented by a 2D rectangle
25 within 3D display model 102. During the rendering process, various well-known techniques, such as ray tracing, are used to map objects from 3D display model 102 into corresponding locations in 2D display 104.

[0035] The bottom portion of FIG. 1 illustrates some of the system components that make it possible to map 2D windows into 3D display model 102 in accordance with an embodiment of the present invention. Referring to FIG. 1, applications 114 and 116 are associated with windows 108 and 110, respectively.

5 A number of components are involved in facilitating this association. In particular, applications 114 and 116 are associated with xclients 118 and 120, respectively. Xclients 118 and 120 in turn interact with xserver 122, which includes an associated xwindow manager. These components work together to render output bitmaps 124 and 126 for applications 114 and 116 to be displayed in

10 windows 108 and 110, respectively. These bitmaps 124 and 126 are maintained within back buffer 128.

[0036] Code module 130 causes bitmaps 124 and 126 to be displayed on corresponding windows 108 and 110. More specifically, code module 130 retrieves bitmap 126 and converts it into a texture 132, which is displayed on the

15 front face of window 110. This is accomplished through interactions with 3D scene manager 134. Bitmap 124 is similarly mapped into window 108.

[0037] 3D scene manager 134 can also receive input from a 2D pointing device, such as mouse 136, and can communicate this input to applications 114 and 116 in the following way. 3D scene manager 134 first receives an input

20 specifying a 2D offset from mouse 136 (step 202). Next, the system uses this 2D offset to move a cursor 109 to a new position (x_1, y_1) in 2D display 104 (step 204).

[0038] The system then determines if cursor 109 overlaps a window in 3D display model 102 (step 206). This can be accomplished by projecting a ray 107 from viewpoint 106 through cursor 109 and then determining if the ray intersects

25 a window. If there is no overlap, the process is complete.

[0039] Otherwise, if there is overlap, the system uses the 3D position (x_2, y_2, z_2) within display model 102 where ray 107 intersects window 110, as well

as attributes of window 110, such as position and rotation attributes, to determine the 2D position (x_3, y_3) of this intersection with respect to a 2D coordinate system of window 110 (step 208). The system then communicates this 2D position (x_3, y_3) to application 116, which is associated with window 110 (step 210).

5 **[0040]** 3D scene manger 134 is also coupled to lookup table 135. Lookup table 135 contains entries specifying relationships between windows. As described later in FIG. 6, lookup table 135 allows 3D scene manager 134 to determine if windows 108 and 110 should be displayed on separate objects, or if they should be displayed on different sides of the same object within 3D display
10 model 102.

[0041] Various user inputs, for example through mouse 136 or a keyboard, can be used to manipulate windows within 3D display model 102.

Rotation Around Viewpoint

15 **[0042]** FIG. 3 presents a flow chart illustrating how input from a pointing device causes objects to rotate around a viewpoint 106 in 3D display model 102 in accordance with an embodiment of the present invention. First, the system receives an input from a 2D pointing device indicating that a rotation is desired (step 302). For example, the system can receive a movement input from mouse
20 136. In response to this input, the system can rotate objects within the 3D display model around viewpoint 106, or alternatively around another point within 3D display model 102 (step 304). This rotational motion makes it easier for a user to identify window boundaries and also gives the user a feeling of depth and space.

Displaying Additional Information on Back of Window

25 **[0043]** FIG. 4A illustrates an exemplary window 401 in 3D display model 102, and FIG. 4B illustrates how window 401 is rotated to display additional

information on the backside of window 401 in accordance with an embodiment of the present invention. Referring to the flow chart in FIG. 4C, the system first receives a command (possibly through a mouse or a keyboard) to rotate window 401 (step 404). In response to this command, the system rotates window 401 so that additional information 402 on the backside of window 401 becomes visible (step 406). Additional information 402 can include application information, such as application version information, application settings, application parameters, application properties, and notes associated with a file or a web page that is displayed in the window. In one embodiment of the present invention, the system allows the user to modify application information 402 on the backside of window 401. This enables the user to change application parameters, if necessary.

[0044] This additional information 402 can also include a window associated with the same application, a window associated with a related application, a window associated with a different application, a modal dialog associated with the application, or a modal dialog associated with the OS.

3D Object with Multiple Viewing Surfaces

[0045] FIG. 5 illustrates 3D object 111 with multiple viewing surfaces in accordance with an embodiment of the present invention. In the orientation shown in FIG. 5, 3D object 111 has 3 visible surfaces, which display window 502, window 504, and window 506. Note that 3D object 111 has additional surfaces that are not visible in the current orientation. Also note that in general, 3D object 111 is not limited to being a slate or a cube, and can be any size or shape, and can have any number of visible surfaces.

25

Displaying a Window

[0046] FIG. 6 presents a flow chart of the process of displaying a window in accordance with an embodiment of the present invention. The process starts when 3D scene manager 134 receives a direction to open a new window in 3D display model 102 (step 602). 3D scene manager 134 looks-up the title of the window to open in lookup table 135 (step 604) and determines if the window title is linked to the title of any of the windows that are currently open within 3D display model 102 (step 606). If the title is not linked, or is not listed in lookup table 105, 3D scene manager 134 opens the window on a new 3D object within 3D display model 102 (step 608). However, if the title is linked, 3D scene manager 134 opens the window on a different surface of the 3D object that is displaying the related window (step 610). Note that displaying the new window on an existing 3D object might result in changing the orientation of the 3D object so that the pre-existing related window is no-longer visible from viewpoint 106.

[0047] Table 1 illustrates an exemplary lookup table 135 in accordance with an embodiment of the present invention.

Front window name	Windows that can be placed on the back
^Web Browser .*\$	^Preferences\$ ^Alert\$
^Editor .*\$	^Preferences\$ ^Save As\$ ^Open\$
^Music Player .*\$	^Note Pad\$
^Photo Viewer .*\$	^Email\$

Table 1

When a request to show a new window is sent to 3D scene manager 134, 3D scene manager 134 first finds the row in lookup table 135 whose "Front window name" matches the currently focused window based on specific regular

expression. Next, 3D scene manager 134 checks if the requested window's title matches to the regular expression shown in the "Windows that can be placed on the back" column. If it matches, 3D scene manager 134 rotates the window by 180 degrees so that the user can see the back side of the window. Finally, 3D scene manager 134 places the requested window on the back side of the window.

Displaying a Modal Dialog

[0048] FIG. 7 presents a flow chart of the process of displaying a modal dialog in accordance with an embodiment of the present invention. The system starts when 3D scene manager 134 is directed to display a modal dialog (step 702) or any other dialog that requires user intervention before operations may continue on an open application. 3D scene manager 134 then determines all of the currently visible windows that are affected by the modal dialog (step 704). Next, 3D scene manager 134 makes affected windows less visible. This can be accomplished by rotating all of the affected windows so that they are no longer visible to viewpoint 106 (step 706). Finally, 3D scene manager 134 displays the modal dialog on the backside of one of the rotated windows (step 708).

Object Translucency

[0049] FIG. 8 illustrates an example where object translucency can be used to facilitate displaying related information in accordance with an embodiment of the present invention. In this example, the 3D interface displays user calendar 802, group calendar 804, and stacked objects 806. Note that stacked objects 806 comprises user calendar 802 placed on top of group calendar 804 as seen from viewpoint 106. When the cursor is moved off of an object in 3D display model 102, the object becomes semi-translucent. This allows an observer to see any object located behind or underneath of the object.

[0050] In the illustrated example, when the user locates the mouse cursor over the top of stacked objects 806, the user will see only user calendar 802. However, when the cursor is moved off of stacked objects 806, objects in stacked objects 806 become translucent, thereby allowing the user to see all of the objects simultaneously. In this instance, appointments (designated by the cross-hatched regions) on both user calendar 802 and group calendar 804 are visible to the user at the same time, and allow for the user to visually detect any calendaring conflicts.

[0051] The foregoing descriptions of embodiments of the present invention have been presented for purposes of illustration and description only. They are not intended to be exhaustive or to limit the present invention to the forms disclosed. Accordingly, many modifications and variations will be apparent to practitioners skilled in the art. Additionally, the above disclosure is not intended to limit the present invention. The scope of the present invention is defined by the appended claims.